PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

B1

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F

A2

(11) International Publication Number: WO 98/54631

(43) International Publication Date: 3 December 1998 (03.12.98)

(21) International Application Number: PCT/US98/07337

(22) International Filing Date: 14 April 1998 (14.04.98)

(30) Priority Data:
08/863,807 27 May 1997 (27.05.97) US

(71) Applicant: MERRILL LYNCH & CO., INC. [US/US]; 250 Vesey Street, New York, NY 10281 (US).

(72) Inventors: SIMMONS, Edward, F.; 138 George Street, Lambertville, NJ 08530 (US). PERU, David, Robert; 123 Kimball Street, Iselin, NJ 08830 (US).

(74) Agent: BOLLINGER, James, M.; Hopgood, Calimafde, Kalil & Judlowe, 60 East 42nd Street, New York, NY 10165 (US).

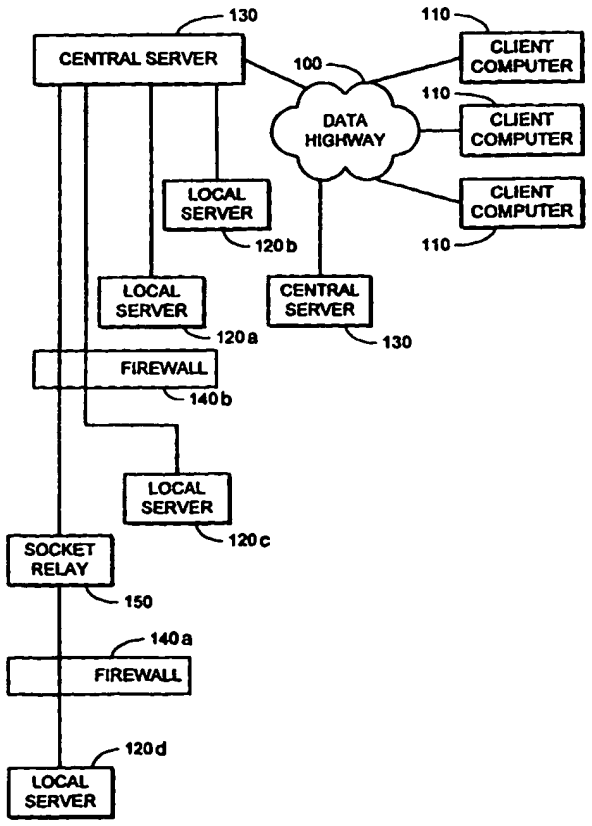
(81) Designated States: AU, CA, CN, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: SYSTEM FOR NETWORK FILE DISTRIBUTION

(57) Abstract

A system having client computers, at least one central server and local servers. The local servers control the updating of content on HTTP farm servers directly under its control. Client computers are used by content providers and system administrators to control the content on the ultimate HTTP servers. The central server receives information from the client computer, allows for testing of the content, if desired, and then forwards the information to the local servers as appropriate. The release and maintenance of content is strictly controlled on several levels.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

System for Network File Distribution

Field of the Invention

This invention relates generally to a system for distributing information to a plurality of computers. More specifically, the invention relates to a system for distributing, updating and maintaining files, originated by various sources, on remote servers subsequently accessed by multiple users over networks.

Background of the Invention

In the past few years, personal computers have grown beyond mere desktop processors to become sophisticated vehicles supporting extended communications and related functionality. Today, workstations are routinely used for linking into vast networks of computer servers to search for information, exchange e-mail, files, video and audio information, and access large pools of remotely stored information in addressable databases. Perhaps the best example of this new function is found with the expanded communication available with the global Internet and more particularly, the World Wide Web segment of the Internet. Through simple graphical interfaces, anyone with a computer may access a practically limitless amount of information, regardless of the information's origin or location.

The financial services industry is taking part in the push to expand the Internet beyond its historical role as a research resource mainly used for the dissemination of academic information and ideas (but lacking in transactional functionality). From online banking to electronic money for purchases (e.g., "E-CASH") to

investment portfolio management, financial services on the Internet are expanding rapidly. As examples, it is already possible with a computer over the Internet to maintain bank accounts, pay bills, track and trade investment assets, and place orders for products.

It is well known that the functioning of the Internet is not controlled from one central location, nor are all the files accessible over the Internet contained in one location. Instead, the Internet is a widely distributed network of servers, gateways and hubs, each either enabling the Internet as a whole (such as by routing data requests and responses) and/or housing data and functions accessible by end users. This distributed scheme of data and functions applies not only to the Internet in general, but also to the individual providers of data and functionality. For example, one organization may not have all of its Internet related files contained on one server, but may have them shared between multiple, geographically separated servers. With larger organizations, each division or subsidiary may have its own dedicated servers to store and maintain only those Internet files particular to its responsibilities. It is also possible for various divisions' files to be stored on a more centralized server, with the responsibility for updating each file falling to the particular division from which it originated. In any case, all of the files for that organization are usually ultimately linked and accessible through the umbrella site of the organization, so that an end user accessing the files is unaware of each file's physical location.

Within organizations, the storage scheme of various files has been changing with the advent of intranets replacing the older dedicated local area networks. For example, there may be some servers solely for internal use (private sites), while other servers or portions of servers are accessible to the Internet and public in general (public sites). Again, there may be various

origination points for the files on any of these servers and also various groups or individuals responsible for maintaining and updating individual files.

Regardless of the use or permitted access to the servers, there is a need to maintain and control the currency of the various files on various servers. To date, several methods have been used. For example, if the files to be updated are all contained on one server, the maintenance of those files could be accomplished by overwriting the older files with new ones contained on a floppy disk or other portable storage medium. However, when there are multiple servers at many different distant sites, the magnitude and difficulty of the task becomes evident. Manually keeping track of all the files and then sending out update disks or tapes, for example, is a difficult, slow, labor-intensive and unreliable method. At some sites, files might be frequently changing, requiring frequent manual delivery of new updates. Alternatively, because the servers are likely all networked in one form or another (for example, through the Internet itself), it is also possible for an operator to individually and manually access each server in turn and overwrite the existing files. However, all of these methods require gathering the necessary changes or updated files from diverse origins, since responsibility for maintaining the files may be fragmented. The gathering step may also become unwieldy, as some or a majority of the information may originate with many different content providers both inside and outside the organization directly responsible for the Internet sites. Thus, information would flow in a multi-step operation, first from each individual content provider in turn to an individual or group within the controlling organization and then each file would be manually transferred to the end-user-accessible server.

Regardless of the exact method used, these manual, labor-intensive methods introduce other difficulties with respect to the version control and history of the content files on various servers, not to mention the possibility for user error. If content files on each server are updated individually and/or manually, keeping track of what has been updated becomes a bookkeeping burden. Further, maintaining or even archiving previous versions of content files only intensifies the problem. Factoring the possibilities of multiple entities being responsible for different groups of related or linked content files, the burden grows.

It is understood that some information has been automatically updated on internet servers in the past. However, this has been limited to frequently or continuously updated data streams, such as stock quotes or news items. This updating does not address the maintenance or control of the underlying server files, including, but not limited to data, page layout and executables, which may, for example, affect the way in which the data streams are displayed or processed for the end user.

Summary of the Invention

In view of the deficiencies of the prior art, it is an object of the present invention to provide a system for updating, controlling and maintaining multiple computer servers simultaneously.

It is another object of the present invention to provide for updating of computer servers from multiple file sources.

It is a further object of the present invention to provide for version control, history and archiving of updated and superseded files.

It is yet another object of the present invention to provide capability for verification and testing of updated source information before it is updated on remote servers.

It is a still further object of the present invention to provide a secure environment in which to accomplish updating of multiple computer servers.

In accordance with these objects, a system is disclosed having client computers, at least one central server and local servers. The local servers control the updating of content on HTTP farm servers directly under its control. Client computers are used by content providers and system administrators to control the content on the ultimate HTTP servers. The central server receives information from the client computer, allows for testing of the content, if desired, and then forwards the information to the local servers as appropriate. The release and maintenance of content is strictly controlled on several levels.

Brief Description of the Drawings

The aforementioned and other objects and advantages will become apparent to those skilled in the art upon reading the following detailed description of the preferred embodiments in conjunction with a review of the appended drawings, in which:

Fig. 1 is an overall system schematic showing a preferred embodiment of the network distribution file system of the present invention;

Fig. 2 is a logic flowchart of a remote client computer accessing the central server of the system of Fig. 1;

Fig. 3 is a logic flowchart of a security feature of the client/central server connection of the system of Fig. 1;

Fig. 4 is a logic flowchart of a remote local server accessing the central server of the system of Fig. 1;

Fig. 5 is a logic flowchart of another security feature of the local/central server connection of the system of Fig. 1; and

Fig. 6 is a schematic diagram showing the location of various files in the system of the present invention.

Detailed Description of the Preferred Embodiments

Referring now to Fig. 1, the overall schematic of a system according to the present invention is shown. In overview, the system allows an organization to make content available to others, such as over the internet, while giving the organization the ability to easily and automatically control that content, though it may be scattered across multiple servers. With respect to this application, the term "client" is anyone, either within or outside the controlling organization who has authorization to alter or provide content (e.g., files, data, etc.) to the system. "User" is a person who ultimately accesses the content, such as in an HTTP environment, and/or takes advantage of the functionality of the content. The user may be within or outside the controlling organization, as described below.

Client computers 110 are accessed by various classes of clients: system administrators, content providers and programmers, to name a few. The client computers provide access to the files of the system for updating or other maintenance purposes. The local servers 120, on the other hand, feed content to their HTTP server farms respectively, as appropriate for ultimate use by users. In other words, the local servers put the content in a position where it may be accessed by anyone with access to that server farm. For example, two of the servers 120a,120b shown are examples of private access servers. These servers would be accessed only by users within the organization controlling the system. These servers 120a,120b could be part of a corporation's intranet, taking the place of a hardwired local area network. Two other servers 120c,120d, however, are outside of the controlling organization and may be accessed by any public user with the appropriate computer capabilities. Of

course, some of the content, such as the user's financial account information, may require passwords or other security features, but at least the connection can be made through public computer networks. For further security purposes, the local servers 120c,120d outside the controlling organization are separated from their respective central servers 130 by firewall systems 140a,140b. Individual firewall systems are generally known. Depending on the location within the internet of a local server, it is contemplated that a socket relay 150, described more fully below, would be needed.

The socket relay 150 is located between the double firewalls 140a,140b. The socket relay prevents certain techniques for bypassing firewalls by altering the address of the data packets and retransmitting the packets between the firewalls 140. For example, the local server 120d attempts to make a connection to 1.1.1.1:7100 (this address is broken down into the server "1.1.1.1" and the listening port "7100"). The firewall 140a is configured to translate the address of the packet into 2.2.2.2:7100, "2.2.2.2" being the address of the socket relay 150 running on a separate computer. The data packets are then placed on the safe side of the firewall 140a and transmitted to the socket relay 150. The socket relay 150 then connects to the second firewall 140b at 3.3.3.3:7100 and transmits the data packets to address 3.3.3.3:7100. The packet address is translated by the second firewall 140b into 4.4.4.4:7100, "4.4.4.4" being the address of the central server 130. The packets are placed on the safe side of the firewall 140b and transmitted to the central server 130. The end result is that a packet sent to 1.1.1.1:7100 arrives at 4.4.4.4:7100 as though it was originally sent to that address. However, by using the socket relay 150, which does not alter the data packets, but transmits them to a different address, attempts to bypass the firewalls are thwarted.

Central servers 130 are situated between the local servers 120 and client computers 110, acting as conduits and gatekeepers for content flow. The central servers 130 form the backbone of the system and include much of the functionality to achieve the system's objects. As seen in Fig. 1, there may be more than one central server 130. Each central server 130 is connected to a plurality of local servers 120, each which either include the files that are subject to updating or have connections to and control over other servers that do.

The specific hardware and connection protocols for the various servers are not critical, although there are some preferred features. The hardware preferably includes the storage capacity to maintain the necessary files of the system, as described below. The central servers have higher storage needs, as they house master copies of all data files under their control. The central servers may also hold archived copies of all previous versions of all files, and thus would require significantly more storage. All of the servers must also include standard hardware and software for making remote connections, such as over telephone lines, satellite or microwave transmissions or hardwiring. The protocol used is immaterial, although the current TCP/IP standard is preferred. The connections between the client and central servers are preferably made over a data highway 100, such as a Novell network within the controlling organization or the Internet itself. The connections between the central and local servers are similar. In the preferred embodiments, the central and local servers are workstations running Microsoft® Windows NT. The client computers are workstations preferably running under either Windows 95 or Windows NT. All of the servers preferably communicate using the TCP/IP protocol. The exact hardware configuration, operating system, and communications protocol of each server is not critical and it is contemplated that improvements in workstations and communications protocols may be incorporated into the present invention without departing from the invention.

The potential variation in the client computers is largely due to the fact that some of those servers may be outside the controlling organization and are thus not subject to control. It is only necessary that those servers be capable of running the client or local server portion, respectively, of the software portion of the present system. The software components of the system are preferably built using Microsoft's Foundation Classes (MFC). Specifically the system uses the MFC CSocket class for streaming complex aggregate data types over TCP/IP connections.

To accomplish connections between a client computer and a central server, the client computer should be pre-configured with the network address (for example, TCP/IP address) of the central server or else the client may input the address. This may also include identifying a specific listening port on the central server. Each central server then maintains a database of those client computers and users that are permitted to access that particular central server. The database also contains, for each user, the types and/or specific files that may be updated and/or deleted by the user. This access information is set by the controlling organization to maintain security. In the preferred embodiment, the client also needs to know the specific local and central servers that form a pathway to the target HTTP server. This address information is then programmed into the client computer. Alternatively, it is contemplated that a database may be used for automatic retrieval by the client computer of the appropriate local and client computer information for a particular selected target HTTP server.

Client/Central Server Connection

The connection sequence between a client computer and a central server is illustrated in the logic schematic of Fig. 2. Assuming that the central server is up and running with its portion of the software of the present invention, that server will be listening on two separate socket ports waiting for connections. One port is listening for connections coming from client computers while the second port is listening for connections coming from local servers. Of course, more than two ports may be used and simultaneous connections may be made, although this raises the possibility of two clients attempting to update the same file at the same time. This is usually avoided in the preferred embodiment by only allowing one client to update an associated target HTTP file. This also helps maintain control over the version of the files, since only one client has the ability to update. If for some reason more than one client is given update access to a file, the client whose update command was received first would be executed, while a near-simultaneous command from another user would result in an error message being sent back to the second client computer and the file not being further updated.

Once a client launches the client portion of the software from their remote location (block 210), a socket connection is initiated by the client computer (block 220) and established with the central server. The central server then acknowledges the connection (block 230). The client computer then sends a login request to the central server, using the username and security password of the client attempting to update the files (block 240). It is contemplated that encryption may be performed on the username and password data to prevent unauthorized users, using such programs as packet "sniffers," to gain improper access to the system. It is further contemplated that some form of encryption and/or authentication may be used for all data passed between the various servers of the

system. Assuming the username and password match that already stored in the database on the central server, the central server responds with the client's account information, which includes, but is not limited to, the following information: file types and specific file names or file prefixes the client is permitted to update; and target local server and/or HTTP farm machines the user is allowed to update (block 250). Based on these restrictions (previously set and maintained by the controlling organization's administrators), the client may now update content files.

After the connection is established, the client computer is the master and the central server is the slave, i.e., the central server waits for commands from the client computer. For security purposes and resource management, to ensure that the client computer is still active and that the socket connection has not died, the central server sends a test packet periodically, even though the central server is the slave in the connection. By using the test packets, the servers can detect when a connection has been lost or dropped, allowing the servers to immediately free up resources previously used by that connection for other connections. As shown in Fig. 3, the central server sends a test packet that contains no data, only control signals (block 310). If the client computer does not echo the packet back (block 320), the central server releases the connection (block 330). If the client computer does successfully echo the packet (block 320), the central server maintains the connection (block 340) and begins waiting a predetermined period (preferably two minutes - block 350) to send another test packet (block 310). It is preferred that the delay at block 350 be large enough so that the test packets do not create a performance or load problem on the network.

Referring now to Fig. 6, the client computer contains various files pertaining to the present invention. Of course, all of the servers in the system will have the necessary operating system and communications files, in addition to any display, storage or other driver files needed for the general functioning of the server computer. With respect to the present invention, the client computer 110 includes the program executable files 610 of the present invention. The server also contains the source content files 620, which are the files that are transferred ultimately to the local servers and the HTTP farms. Internal data files 630 store information on the central and local servers to which the client computer has access, in addition to any restrictions on access to those servers. Finally, the client computer includes a log file 640, which records all client activity. This can be useful to an administrator in diagnosing a client error, or in recreating what a client desired to do, but was unsuccessful.

The central servers each contain program executable files 650 for their own use. These are in addition to those executable files that may be transferred to local servers at another time. The central servers also each contain a set of master content files 660, which are master copies of the most recent versions of all files that were transferred to any of the local servers connected to each particular central server. Archive data and archived files 670 include a record of all transactions that have occurred to the local servers, as well as copies of all older versions of superseded content files. It is not necessary for all of the archived files to be maintained on the central servers. Preferably, the older archived files are transferred to less accessible media, such as removable tape cartridges, rather than on the servers' own drives. Using the archive data and files, the system is capable of not only recreating the

current configuration of any local server (using master files), but the configuration at any time in the past as well. For example, if a local server is damaged in some way, or there is a data loss on that server, the system can read the archive data to determine which of the current master files had been transferred by the central server and not superseded since transfer. The central server can then transfer all of these content files, including the necessary local server executable files, to completely rebuild the server. For content that may be sensitive, either from a historical viewpoint, or perhaps a legal viewpoint, having older copies of all files and a record of when they were available could be invaluable for resolving future discrepancies.

The central server also contains database files 680 relating to the client and local servers that may access it. For client computers, these files would include username and password information for clients, while for local servers, it would include the machine group names of servers for validation. The database files also include tables detailing which of the master content files are intended for particular local servers. The tables include entries for each master content file and include such information as the following: time/date of last update, and appropriate local servers for transfer.

The local servers 120 contain the program executable files 690 needed to run the system on that computer. The local servers also contain the content files 700 that have been transferred through the central servers 130, as well as connection information files 710, including the network address of the assigned central server and the machine group name, discussed below.

In operation, the client would make desired changes to source content files contained on the client computer. This could involve a single change to a single file, multiple changes to multiple files, additions and deletions of entire files, or any combination of those.

Not all clients are given complete functionality, so it is possible that a particular client would not be permitted to delete files, but merely to add or alter them. Once the desired changes are made to the source files, the client initiates (such as by pressing a screen button) the connection between the client and central servers as discussed above. Assuming the particular client has no restrictions with respect to the updated files, the master content files may then be superseded in one of two modes: replace-all or replace-since a certain date and time. With the replace-all mode, every file in the client's source file directory is copied to the central server. As each file is copied, the previous version stored on the central server is moved to an archive location and a record is made of the transfer. The replace-all mode, however, has the possibility of replacing a relatively new version of a file with an older one that just may happen to be in the user's directory along with new updated files. Therefore, the preferred mode is the replace-since mode, in which the client computer only transfers those files that have been added or updated since a particular date and time. For example, only files updated since the last connection to the central server could be transferred. This greatly reduces the chance of overwriting a newer file. It is also contemplated that the central server could respond to the client computer with a warning when the time/date stamp on a file to be transferred is before the time/date stamp on its current master file. The transfer would only then take place upon specific client authorization.

Each file is also transmitted with additional data packets that include the addresses and directories of the local servers to which the file should be transferred by the central server. The additional data packets preferably also include the network address and local directories of the target HTTP servers to which the file is ultimately transferred. Once all files have been

transferred, the client computer displays the status of the transfer to confirm that all files have indeed been transferred. Failed transfers can be immediately seen and dealt with by the client. The user may also login again with a different username and password, if perhaps the user has multiple sets of content files under its control, or drop the connection to the central server. Once the connection is dropped, the central server resumes its listening state for the next client computer to connect.

The central server also includes a test server, which is preferably a separate process running on the central server machine and configured as an HTTP server. The test server may also be a dedicated separate server machine. When any files are updated, whether they are executable or content files, the client may also transfer the files to the test server. The client may then access the updated files to check the quality of the files. For example, if the updated content files include a web page in HTML or another browser format, the client could access the file on the test server using a browser, including all the file's embedded links to such other files as inline graphics. This is performed to catch any bugs or errors in the updated files before they are distributed by the central servers to local servers, where they are often immediately available to the users in and out of the controlling organization. This test server provides instantaneous feedback to the client regarding possible inoperative sites in their internet user environment.

Central/Local Server Connection

The connection between the central and local servers is somewhat different than the client/central connection in that it is not client-initiated, but is preferably accomplished automatically. As discussed above, the central server, once running, will be listening on at least one port for connections from a local server. Each local server is pre-programmed with the network address (e.g., TCP/IP address) of its assigned central server.

The local server attempts to connect to its assigned central server periodically (block 410), preferably every two minutes. If the connection is successful (block 420), the central server will acknowledge the connection (block 430). If the connection is not successful for any reason (for example, the port may be busy connected to another local server), the local server waits a pre-programmed delay (block 440), preferably two minutes, and attempts to connect again.

Assuming the connection has been made and acknowledged, the local server will then send its machine group name, which is a registry set value (block 450). If the central server database has been pre-programmed with the machine group name for that server (block 460), then the connection is maintained and content file updates will be received by the local server (block 470). If the machine group name is not valid compared to the central server database, the connection is dropped by the central server (block 480).

Unlike the client computer/central server connection, even though the local server initiated the connection, the central server becomes the master, while the local server becomes the slave waiting for commands. However, similar to the client computer/central server connection, the slave server (in this case the local server) sends a test packet to the master (central) for resource management and to know when to try to re-establish a connection to the central server as follows: as shown in Fig. 5, the local server sends a test packet that contains no data, only control signals (block 510). If the central server does not echo the packet back (block 520), the local server releases the connection (block 530) and reclaims the connection resources. If the central server does successfully echo the packet (block 520), the central server maintains the connection (block 540) and the local server begins waiting a predetermined period (preferably two minutes - block 550) to send another test packet (block 310). It is preferred that the delay at block 550

be large enough so that the test packets do not create a performance or load problem on the network. If the connection is lost or dropped unexpectedly (for example, due to power outage), the local server will attempt to re-establish the connection every two minutes until successful.

In the preferred embodiment, each client controls the updating of content files it is authorized to alter. Thus, after the client transfers updated files to a central server and tests them, if desired, the central server in turn transfers the updated files to the connected local servers. In this way, the client maintains full version control on the content files. For example, if the client is updating files on several different target HTTP servers and one is unsuccessful for an unknown reason, the client can immediately restore the other HTTP servers to their previous state using archived files and determine the problem. Then the client would re-transfer the files through the central server to the HTTP servers. The local server does not maintain copies of the older versions of content files, as these are archived on the central server as described above. It is contemplated that the client software includes the option of automatic periodic updates to the central and local servers until all files on the target servers match the latest versions on the client computer.

Upon completion of the file transfer, the HTTP server may need to be stopped and restarted for the updates to take effect. If the client had selected that option (discussed below) when transferring the files from the client computer, the central server will cause the HTTP server to be stopped and then restarted.

Commands of Client computer:

The following are an illustrative list of commands available to a client running the client computer software. These commands facilitate the overall functioning of the system, although they are not all directly involved in the update process.

Delete Remote Files: Allows clients with proper authorization to delete files on the central server.

Freeze Target Machines: Allows clients with proper authorization to freeze target central or local servers to prevent them from receiving any further updates during routine maintenance or another updating sequence.

Ping Selected Servers: Allows clients to test whether or not the target server(s) are up and functioning properly.

Pinch Selected Servers: Allows users to test whether a certain TCP/IP based server is up and running by connecting and disconnecting from the servers's listening port.

Stop/Start NT Servers: In order to update internet site system files, it is sometimes necessary to shut down the HTTP server. Once the new system files have been updated, the HTTP server can be restarted. This option provides the stop/start functionality needed to accomplish this task.

Update System Software: Allows an administrator to update new versions of the system software from a remote location.

Get Software Versions: Allows an administrator to verify the running versions of the central and local server system files from a remote location.

View Targets: Allows a client to see all target machines available to them. From the list, the client may then select which of the local servers it wishes to update.

Delete Target: Allows a client to delete all files and directories at a target location.

Thus it can be seen that the present invention provides a system for updating content and other files on multiple remote servers simultaneously. The system also provides automatic archiving and recovery capabilities, testing before distribution, and strict control on client access to content files, among other features. Of course, various modifications to the system will be apparent to those skilled in the art without departing from the spirit of the invention and these modifications are contemplated as well.

It is to be understood that the embodiments shown and described above, while fully capable of achieving the objects and advantages of the present invention, are shown and described only for the purposes of illustration and not for the purpose of limitation, the invention being only limited by the claims, as follows:

What is claimed is:

1. A system for maintaining files accessible by users over a network, comprising:

a local server, said local server housing content files that are accessible by said users through a computer;

a client computer, said client computer housing source copies of said content files, said client computer permitting a client to update said source content files;

a central server, said central server housing master copies of said content files;

wherein said local server automatically connects to said central server such that files may be transferred from said central server to said local server; and

wherein said client computer connects to said central server and copies updated content files to said central server and said central server subsequently copies said files to said local server.

2. The system as in claim 1, wherein said central server automatically archives older copies of said updated content files upon transfer from said client computer.

3. The system as in claim 1, further comprising a test server, said test server being accessible by said client computer for executing updated content files on said central server before said updated files are copied to said local server.

4. The system as in claim 1, wherein said client computer initiates the connection to said central server, which then communicate in a master - slave relationship, respectively.

5. The system as in claim 4, wherein said central server periodically sends test packets to said client computer while connected and said central server reclaims connection resources if said test packets are not echoed by said client computer.

6. The system as in claim 1, wherein said local server initiates the connection to said central server, which then communicate in a slave - master relationship, respectively.

7. The system as in claim 6, wherein said central server periodically sends test packets to said local server while connected and said central server reclaims connection resources if said test packets are not echoed by said local server.

8. The system as in claim 7, wherein when the connection between said central server and said local server is terminated, said local server will reclaim connection resources and periodically attempt to reestablish a connection to said central server.

9. The system as in claim 5, wherein said client computer will notify the client if said test packets are not echoed.

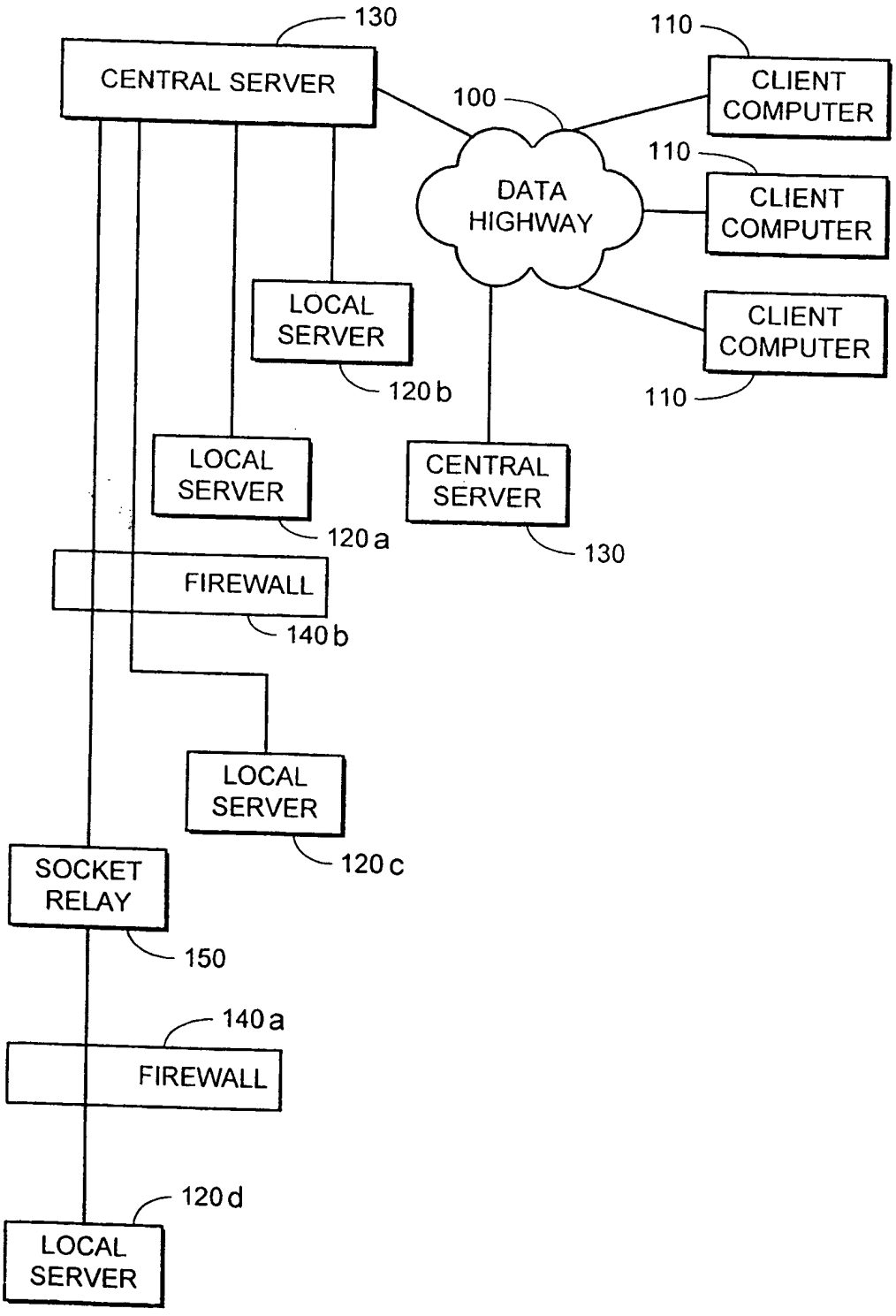

130
CENTRAL SERVER
110
CLIENT COMPUTER
100
DATA HIGHWAY
110
CLIENT COMPUTER
LOCAL SERVER
120 b
CLIENT COMPUTER
110
LOCAL SERVER
120 a
CENTRAL SERVER
130
FIREWALL
140 b
LOCAL SERVER
120 c
SOCKET RELAY
150
140 a
FIREWALL
120 d
LOCAL SERVER

FIG. 1

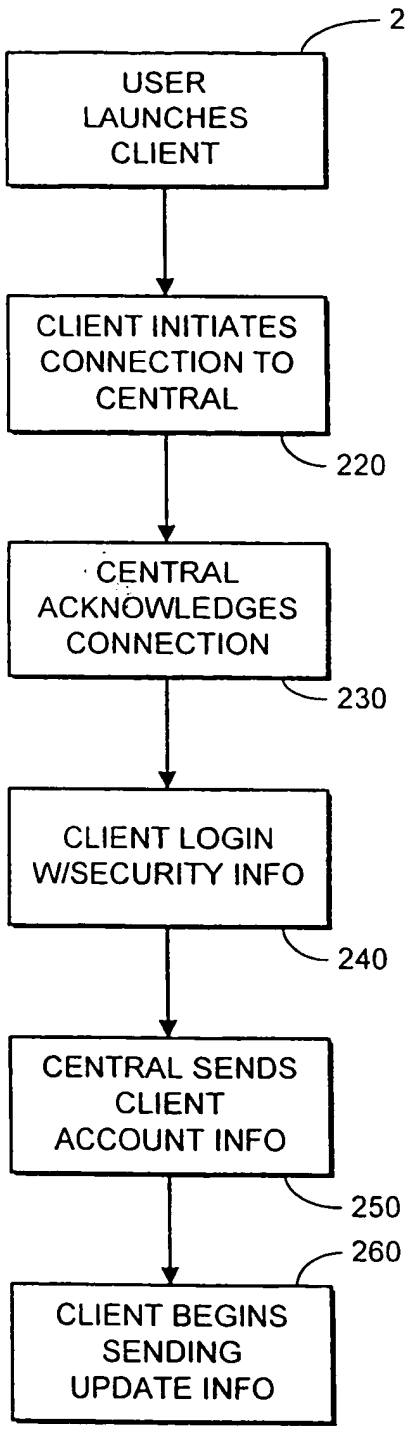

210
USER LAUNCHES CLIENT
CLIENT INITIATES CONNECTION TO CENTRAL
220
CENTRAL ACKNOWLEDGES CONNECTION
230
CLIENT LOGIN W/SECURITY INFO
240
CENTRAL SENDS CLIENT ACCOUNT INFO
250
260
CLIENT BEGINS SENDING UPDATE INFO

FIG. 2

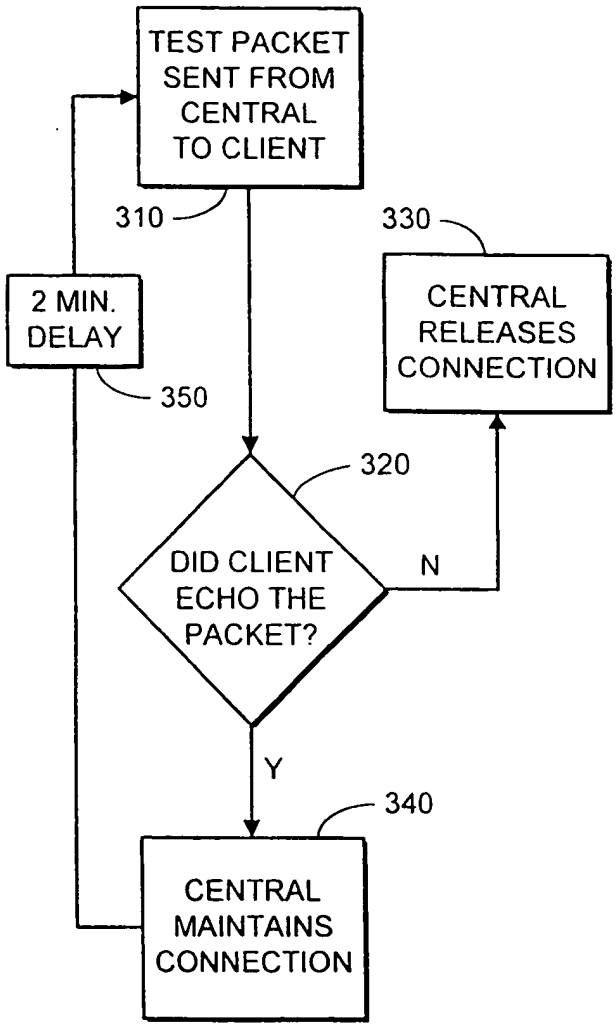

TEST PACKET SENT FROM CENTRAL TO CLIENT
310
330
CENTRAL RELEASES CONNECTION
2 MIN. DELAY
350
320
DID CLIENT ECHO THE PACKET?
N
Y
340
CENTRAL MAINTAINS CONNECTION

FIG. 3

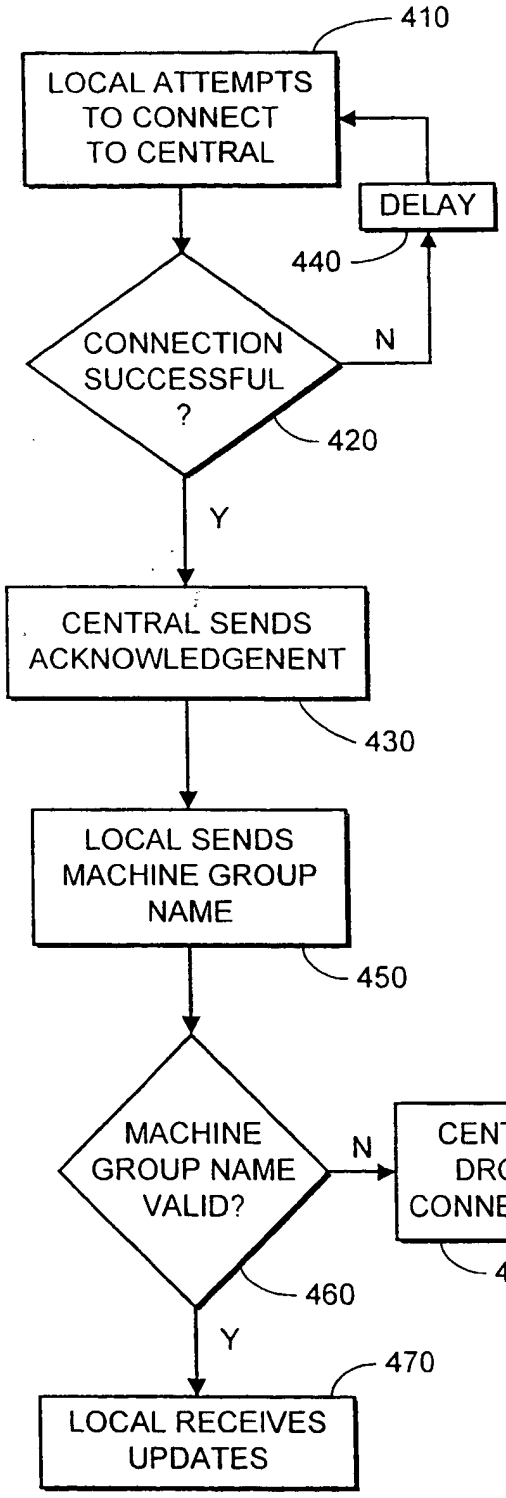

410
LOCAL ATTEMPTS TO CONNECT TO CENTRAL
DELAY
440
CONNECTION SUCCESSFUL ?
N
420
Y
CENTRAL SENDS ACKNOWLEDGENENT
430
LOCAL SENDS MACHINE GROUP NAME
450
MACHINE GROUP NAME VALID?
N
CENTRAL DROPS CONNECTION
480
460
Y
470
LOCAL RECEIVES UPDATES

FIG. 4

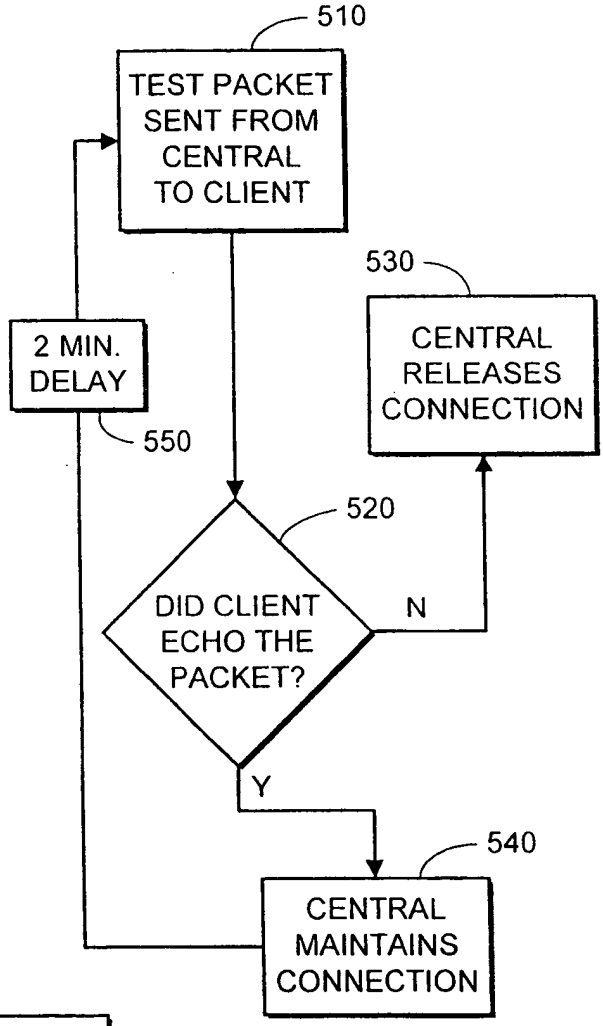

510
TEST PACKET SENT FROM CENTRAL TO CLIENT
530
CENTRAL RELEASES CONNECTION
2 MIN. DELAY
550
520
DID CLIENT ECHO THE PACKET?
N
Y
540
CENTRAL MAINTAINS CONNECTION

FIG. 5

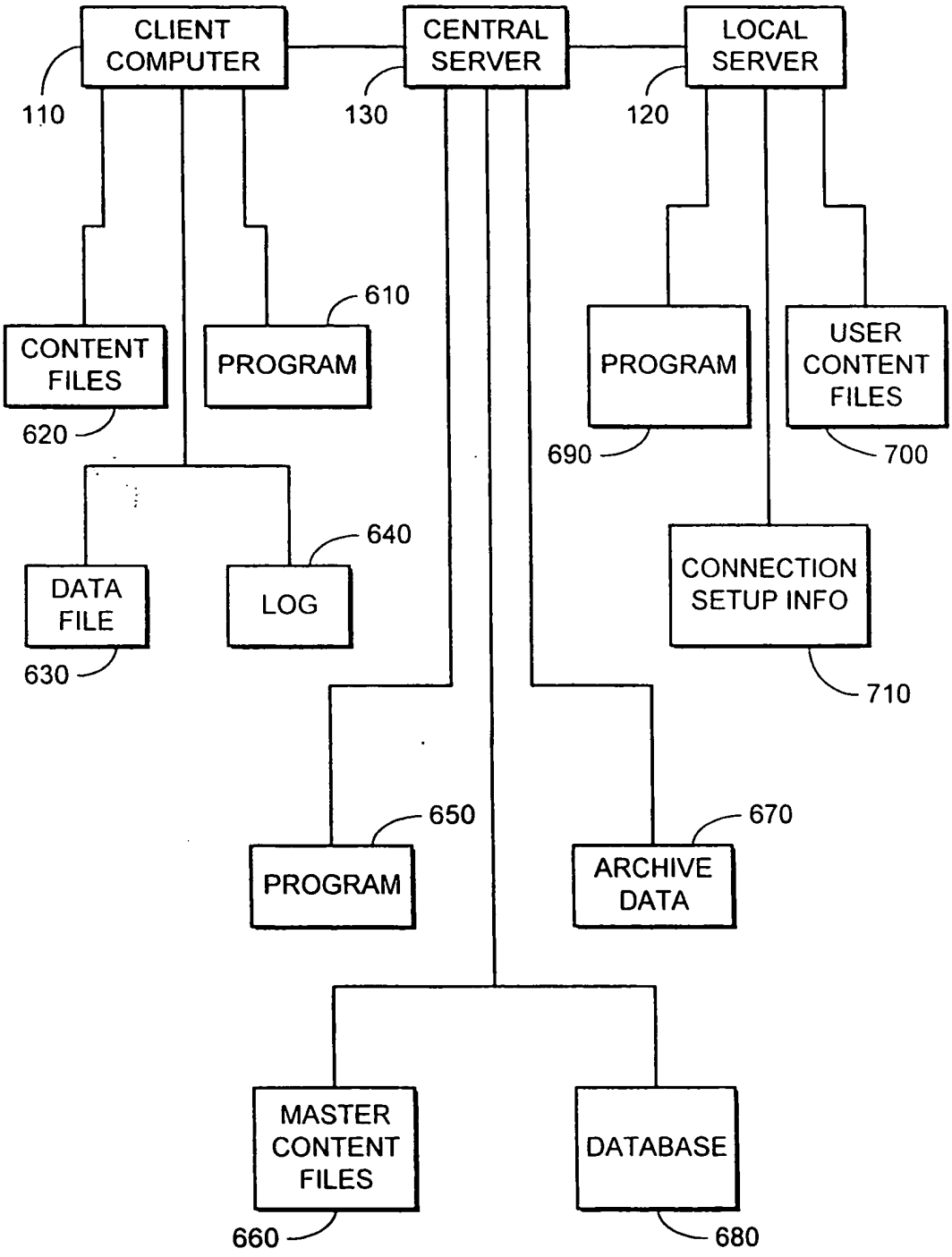

CLIENT COMPUTER
110
CENTRAL SERVER
130
LOCAL SERVER
120
CONTENT FILES
620
610
PROGRAM
PROGRAM
690
USER CONTENT FILES
700
DATA FILE
630
640
LOG
CONNECTION SETUP INFO
710
650
PROGRAM
670
ARCHIVE DATA
MASTER CONTENT FILES
660
DATABASE
680

FIG. 6

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/30 | A3

(11) International Publication Number: WO 98/54631

(43) International Publication Date: 3 December 1998 (03.12.98)

(21) International Application Number: PCT/US98/07337

(22) International Filing Date: 14 April 1998 (14.04.98)

(30) Priority Data:
08/863,807 27 May 1997 (27.05.97) US

(71) Applicant: MERRILL LYNCH & CO., INC. [US/US]; 250 Vesey Street, New York, NY 10281 (US).

(72) Inventors: SIMMONS, Edward, F.; 138 George Street, Lambertville, NJ 08530 (US). PERU, David, Robert; 123 Kimball Street, Iselin, NJ 08830 (US).

(74) Agent: BOLLINGER, James, M.; Hopgood, Calimafde, Kalil & Judlowe, 60 East 42nd Street, New York, NY 10165 (US).

(81) Designated States: AU, CA, CN, JP, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

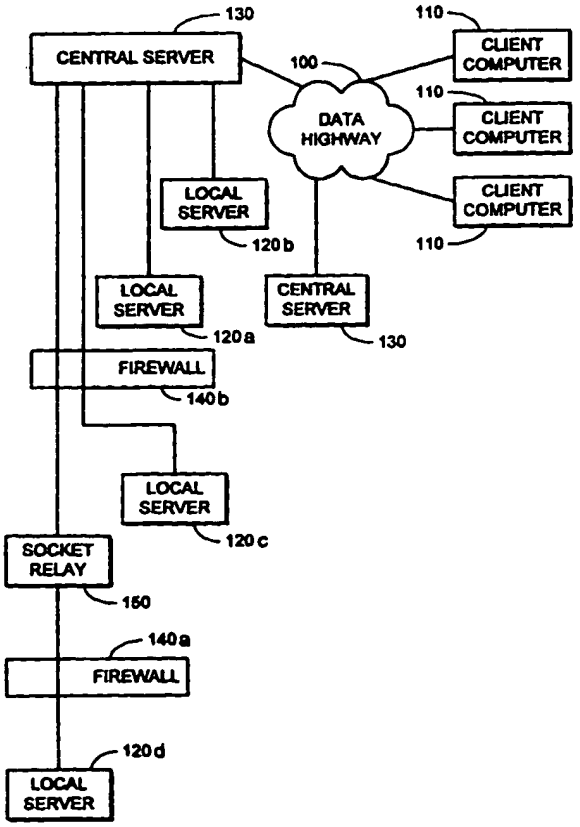
Published
With international search report.

(88) Date of publication of the international search report:
11 March 1999 (11.03.99)

(54) Title: SYSTEM FOR NETWORK FILE DISTRIBUTION

(57) Abstract

A system having client computers (110), at least one central server (130) and local servers (120, a-d). The local servers (120, a-d) control the updating of content on HTTP farm servers directly under their control. Client computers (110) are used by content providers and system administrators to control the content on the ultimate HTTP servers. The central server (130) receives information from the client computer (110), allows for testing of the content, if desired, and then forwards the information to the local servers (120, a-d) as appropriate. The release and maintenance of content is strictly controlled on several levels.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/07337

A. CLASSIFICATION OF SUBJECT MATTER
IPC(6) :G06F 17/30
US CL :707/10, 204; 395/200.33
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 707/10, 204; 395/200.33

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X,P ---- Y,P	US 5,732,219 A (BLUMER ET AL) 24 MARCH 1998, SEE THE WHOLE DOCUMENT.	1,3,4,6 --------- 2,5,7-9
A,P	US 5,737,539 A (EDELSON ET AL) 07 APRIL 1998, SEE THE WHOLE DOCUMENT.	1-9
A,P	US 5,734,823 A (SAIGH ET AL) 31 MARCH 1998, SEE THE WHOLE DOCUMENT.	1-9
A,P	US 5,724,575 A (HOOVER ET AL) 03 MARCH 1998, SEE THE WHOLE DOCUMENT	1-9

[X] Further documents are listed in the continuation of Box C. [ ] See patent family annex.

*	Special categories of cited documents	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search	Date of mailing of the international search report
22 SEPTEMBER 1998	17 NOV 1998
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer MARIA N VON BUHR Telephone No. (703) 305-3800

Form PCT/ISA/210 (second sheet)(July 1992)*

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/07337

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5,530,808 A (HAMMOND et al) 25 June 1996, see the whole document.	1-9
X,P ---- Y,P	US 5,684,984 A (JONES et al) 04 November 1997, see the whole document.	1,4,6 -------- 2,5,7-9
Y,P	US 5,729,735 A (MEYERING) 17 March 1998, see the abstract.	2

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/07337

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

APS, DIALOG
search terms: client/server, central/local/remote, network, copy/archive/backup, files/records/data, updating/synchronizing/resynchronizing, master/slave, testing/ping/echo, connections, communication